WHAT IS CLAIMED IS:

1. A method for emulating in a host computer system a guest computer system, the host computer system including a processor and a host operating system, comprising the steps of:

running an emulator program on the host system that emulates the operation of a guest operating system;

reading in from the processor of the computer system a processor setting associated with the host operating system;

writing the processor setting to memory;

writing to the processor a guest processor setting associated with the emulator program;

wherein the host operating system is logically decoupled from the processor for the processor function related to the guest processor setting associated with the emulator program.

2. The method for emulating in a host computer system a guest computer system of claim 1,

wherein the processor setting associated with the emulator program is an exception vector; and

further comprising the step of calling the exception handler pointed to by the exception vector.

3. The method for emulating in a host computer system a guest computer system of claim 1, further comprising the steps of,

reading in from memory the processor setting associated with the host operating system;

writing to the processor the processor setting associated with the host operating system, wherein the host operating system is logically coupled to the processor for the function related to the processor setting associated with the host operating system.

4. The method for emulating in a host computer system a guest computer system of claim 1, further comprising the steps of:

temporarily logically recoupling the host operating system to the processor for the purpose of performing a function requested by the emulator program; and

logically recoupling the emulator program to the processor for the function related to the processor setting associated with the emulator program following the completion by the host operating system of the function requested by the emulator program.

5. The method for emulating in a host computer system a guest computer system of claim 1, wherein the emulator program operates as an application program on the host operating system.

6. The method for emulating in a host computer system a guest computer system of claim 1,

wherein the processor setting associated with the emulator program is an interrupt routine pointer; and

further comprising the step of calling the interrupt routine pointed to by the interrupt routine pointer.

7. The method for emulating in a host computer system a guest computer system of claim 1,

wherein the processor setting associated with the emulator program is a pointer to an exception vector table; and

further comprising the step of accessing the exception vector table using the processor setting associated with the emulator program.

8. The method for emulating in a host computer system a guest computer system of claim 1,

wherein the processor setting associated with the emulator program is a pointer to a page table associated with the guest computer system; and

further comprising the step of accessing the page table of the guest computer system using the processor setting associated with the emulator program.

9. A method for logically decoupling a host operating system of a computer system from the processor of the computer system, comprising the steps of:

running as an application program on the host computer system an emulator program that emulates the operation of a guest operating system, the guest operating system having a set of functionality associated with the supervisor state of the processor of the computer system;

replacing in the processor as part of a first replacement step a processor setting associated with the functionality of the guest operating system and the supervisor state of the processor such that the host operating system is logically decoupled from the processor with respect to the processor operations associated with the replaced processor setting; and

replacing in the processor as part of a second replacement step a processor setting associated with the functionality of the host operating system such that the host operating system is logically recoupled to the host operating system with respect to the processor operations associated with the replaced processor setting.

10. The method for logically decoupling a host operating system of a computer system from the processor of the computer system of claim 9, wherein the processor setting associated with the functionality of the guest operating system is a pointer to an exception handler provided by the guest operating system.

11. The method for logically decoupling a host operating system of a computer system from the processor of the computer system of claim 9, wherein the processor setting associated with the functionality of the guest operating system is a pointer to a page table of the guest operating system.

12. The method for logically decoupling a host operating system of a computer system from the processor of the computer system of claim 9, further comprising the steps of:

temporarily logically recoupling the host operating system to the processor for the purpose of performing a function requested by the emulator program; and

logically recoupling the emulator program to the processor for the processor settings associated with the guest operating system following the completion by the host operating system of the function requested by the emulator program.

13 The method for logically decoupling a host operating system of a computer system from the processor of the computer system of claim 9 wherein the first replacement step comprises the steps of,

reading in from an identifiable register of the processor a processor setting associated with the host operating system; and

writing the processor setting read in from memory to memory.

14. The method for logically decoupling a host operating system of a computer system from the processor of the computer system of claim 13, wherein the second replacement step comprises the steps of,

reading in from memory the processor setting associated with the host operating system; and

writing the processor setting to the identifiable register of the processor.

15. The method for logically decoupling a host operating system of a computer system from the processor of the computer system of claim 13, wherein the memory is RAM memory.

16. The method for logically decoupling a host operating system of a computer system from the processor of the computer system of claim 9, wherein the processor setting associated with the functionality of the guest operating system is a pointer to an exception handler provided by the guest operating system.

17. A method for rerouting calls by a processor of a computer system to a guest operating system being emulated on a host operating system of the computer system, comprising the steps of:

reading in a first set of processor settings from the processor, the processor settings being associated with functionality provided by the host operating system;

storing the first processor settings in memory;

writing in a second set of replacement processor settings to the processor, the replacement set of processor settings associated with the functionality provided by the guest operating system, wherein with respect to the functionality associated with the replacement set of processor settings, the host operating system is logically decoupled from the processor and the guest operating system is logically coupled to the processor.

18. The method for rerouting calls by a processor of a computer system to a guest operating system being emulated on a host operating system of the computer system of claim 19, further comprising the steps of,

reading in the first set of processor settings stored in memory; and

writing the first set of processor settings to the processor, wherein with respect to functionality associated with the first set of processor settings, the host operating system is logically coupled to the processor.

19. A method for logically decoupling a host operating system from a processor of a computer system, comprising the steps of:

saving the contents of a set of registers on the processor to memory, the set of registers storing at least a part of the processor settings of the processor; and

writing to the set of registers a replacement set of processor settings associated with functionality provided by a guest operating system that resides as an application program on the host operating system.

20. The method for logically decoupling a host operating system from a processor of a computer system of claim 19, wherein the replacement set of registers includes pointers to exception handler routines provided by the host operating system.

21 The method for logically decoupling a host operating system from a processor of a computer system of claim 19, wherein the replacement set of registers includes pointers to interrupt handler routines provided by the host operating system.

22. The method for logically decoupling a host operating system from a processor of a computer system of claim 19, wherein the replacement set of registers includes a page table associated with the host operating system.